Understanding PC-based computer subsystems to maximize total system performance

Jin Guojun

Lawrence Berkeley National Laboratory

Oct. 17, 2003

Build Muscle to Run Faster

- Engineering Design Issues — Understanding systems
  - Hardware system
    - memory sub-system

      cache
      direct memory access (DMA)
    - I/O sub-systems
    - interrupt
    - network system
  - Operating systems
    - system calls
    - system timer
    - context switch
    - berkeley packet filter

Muscle Continued

- Applications
  - network

- Storage
  - disk and disk controller
  - Multi-I/O balance

- Symmetric multi processor (SMP)

- Cost effective price line for buying hardware.

- Silicon Design Evolution
- Network Measurement Technology

Note:

Network measurement is done asynchronously from *receiving* the 1st bit to receiving the last bit.

The first bit does not mean the first bit in a packet; it means the first bit to measure

All other measurements are done synchronously from *sending* the 1st bit to receiving the last bit.

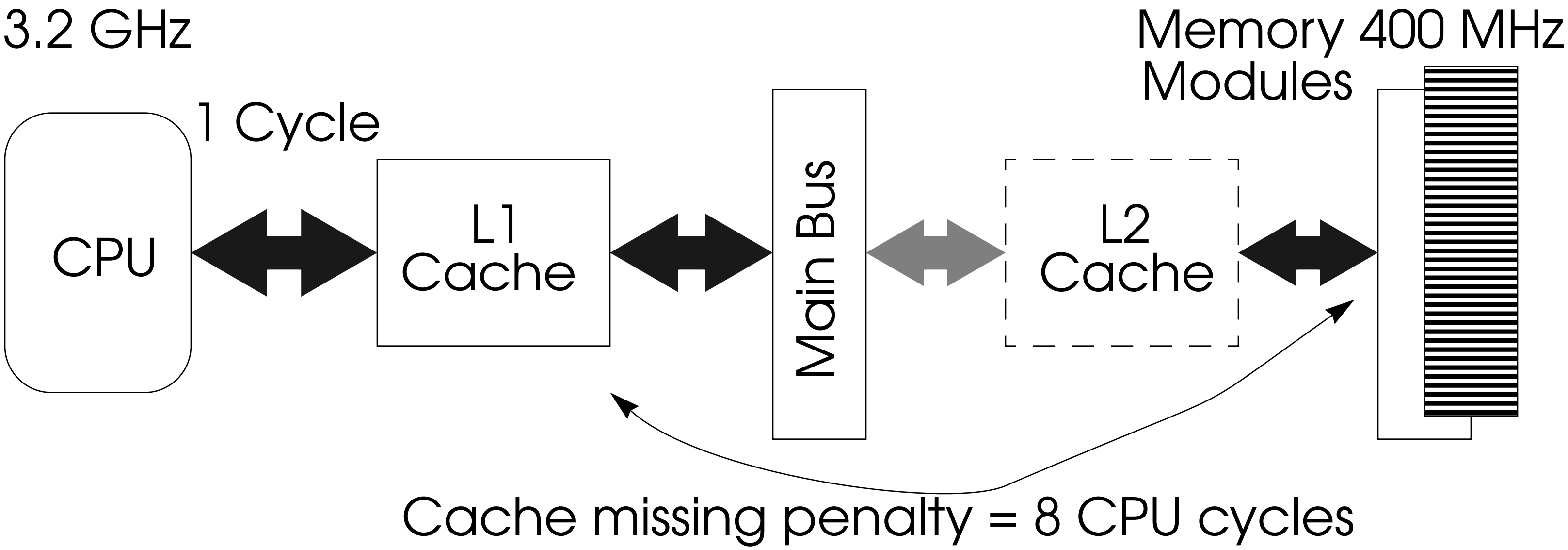
However, around 2008, hardware will be measured with mixed synchronous and asynchronous methods.

Memory Bandwidth

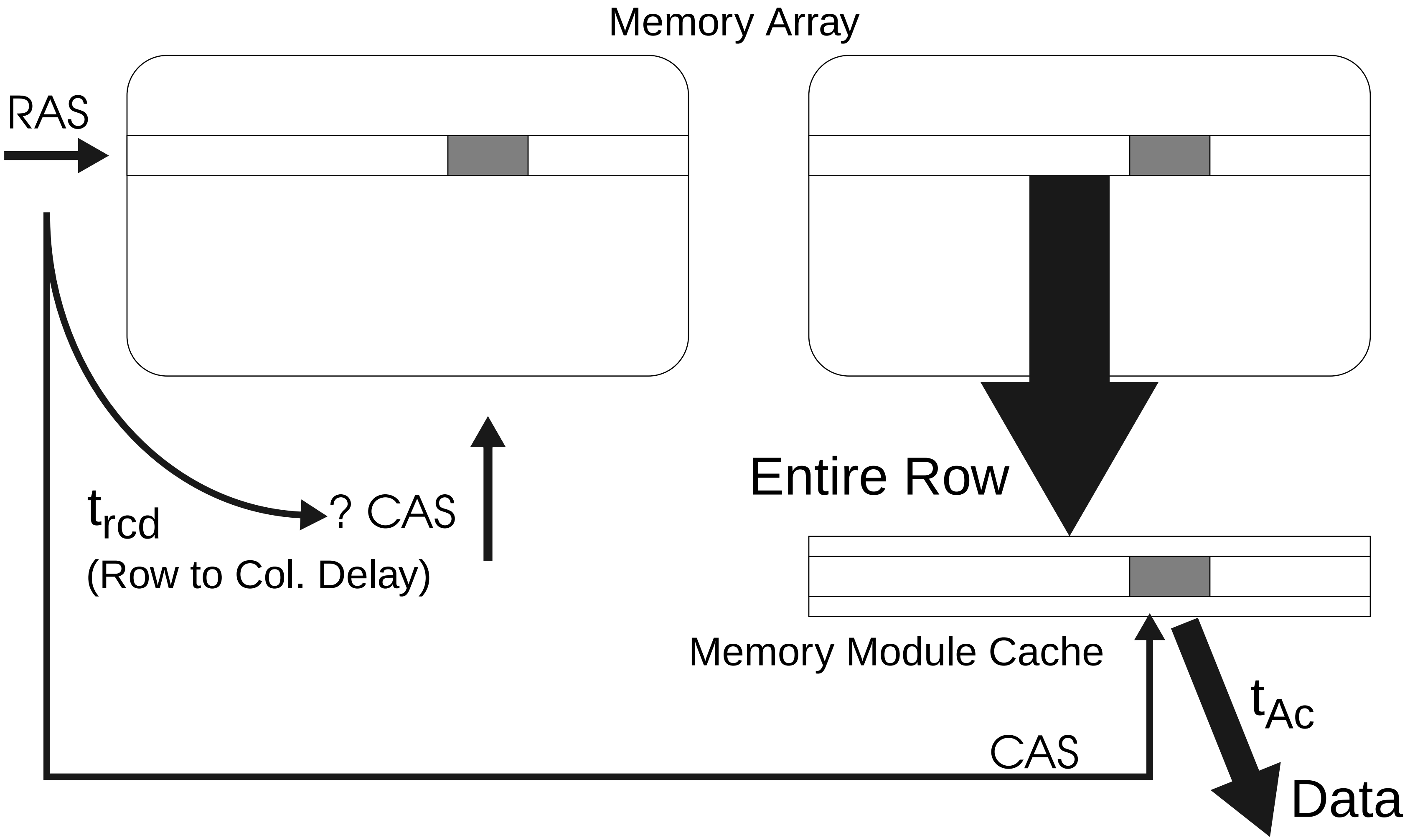
$$\text{MemoryBandwidth} \neq \text{BusClockRate} \times \text{BusWidth}$$

64-bit (8 bytes) 400 MHz memory system does **NOT** produce 3.2 GB/s memory bandwidth:

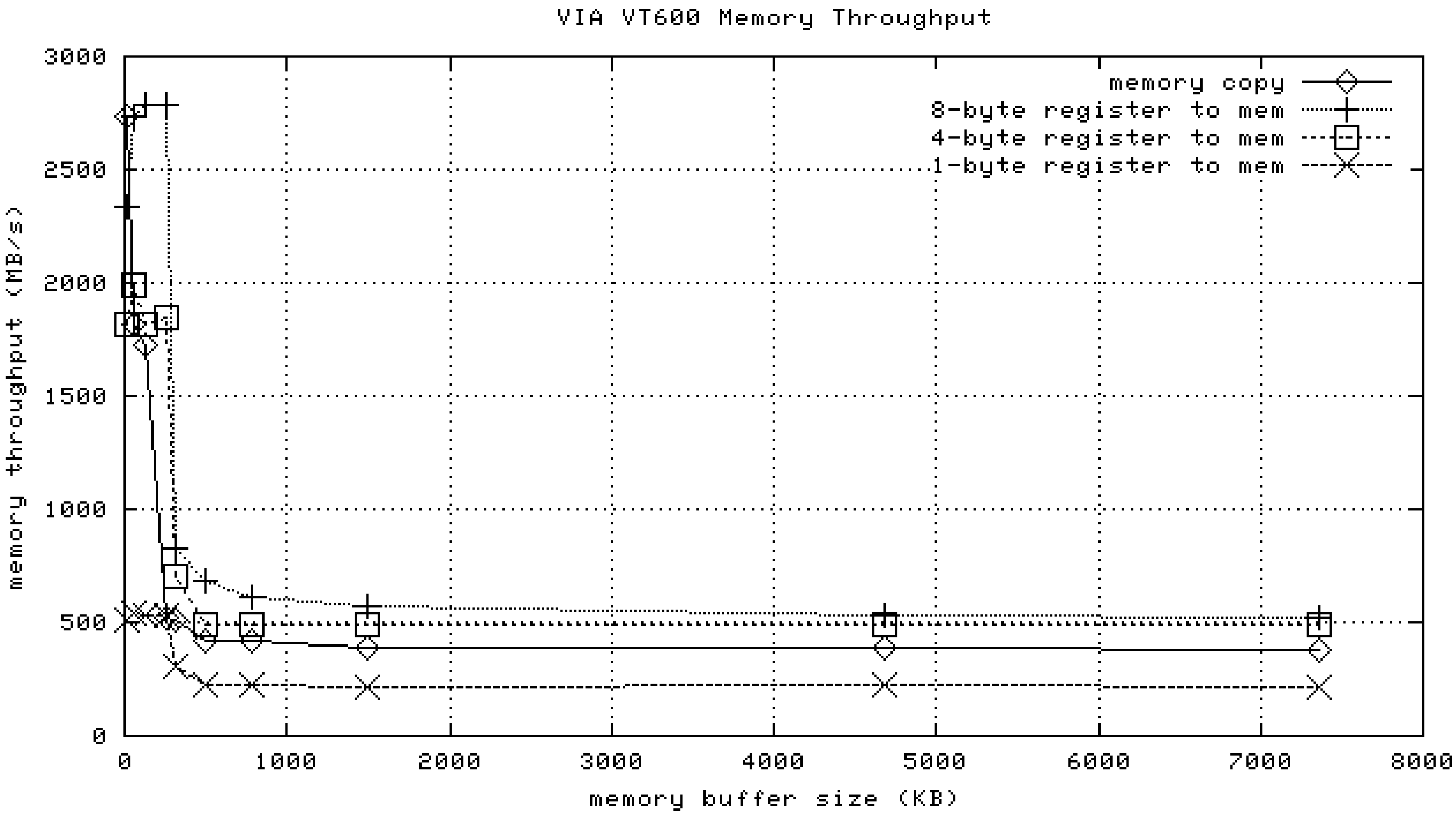
This is because of cache, memory and I/O controllers



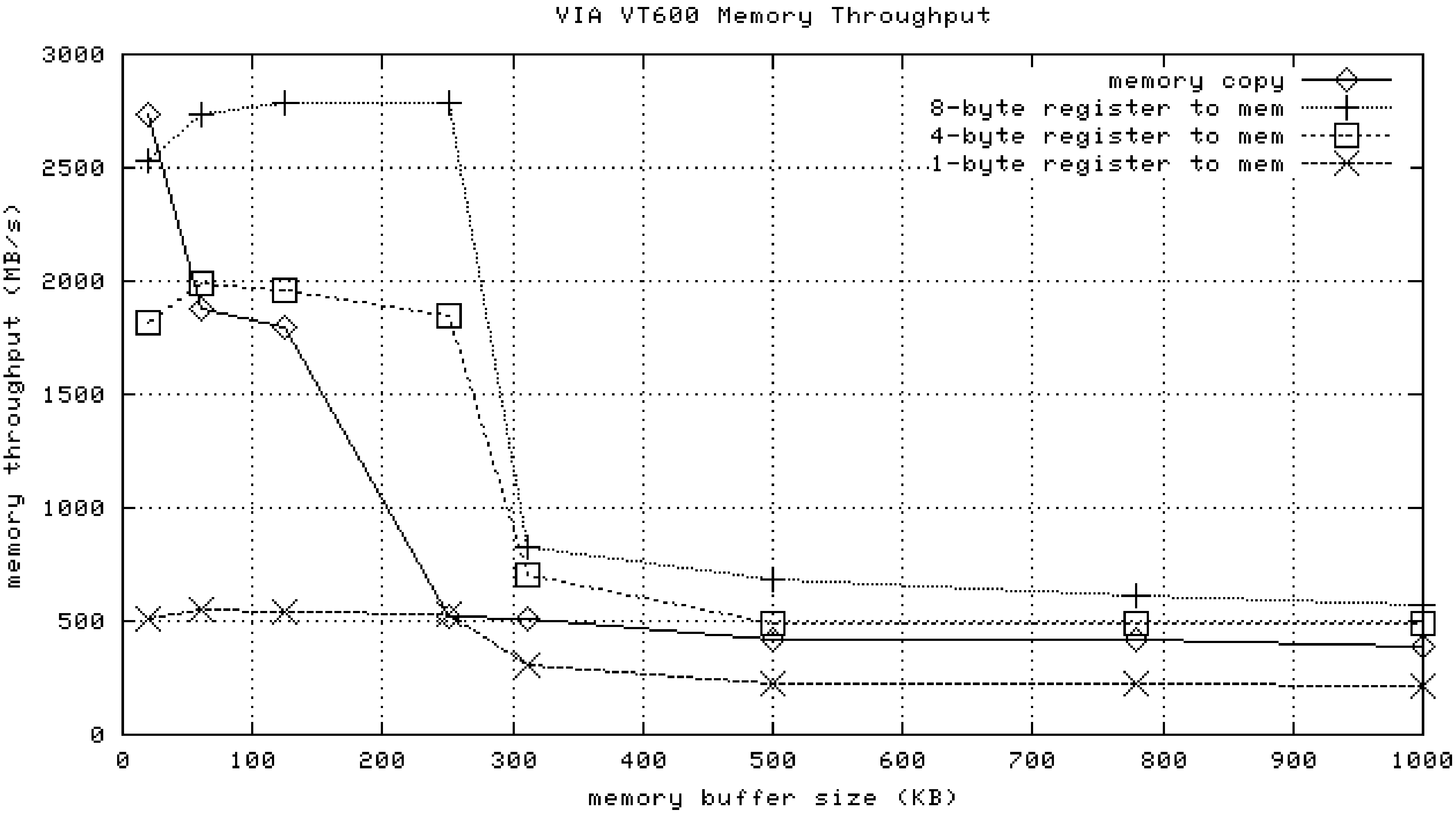
Memory Subsystem



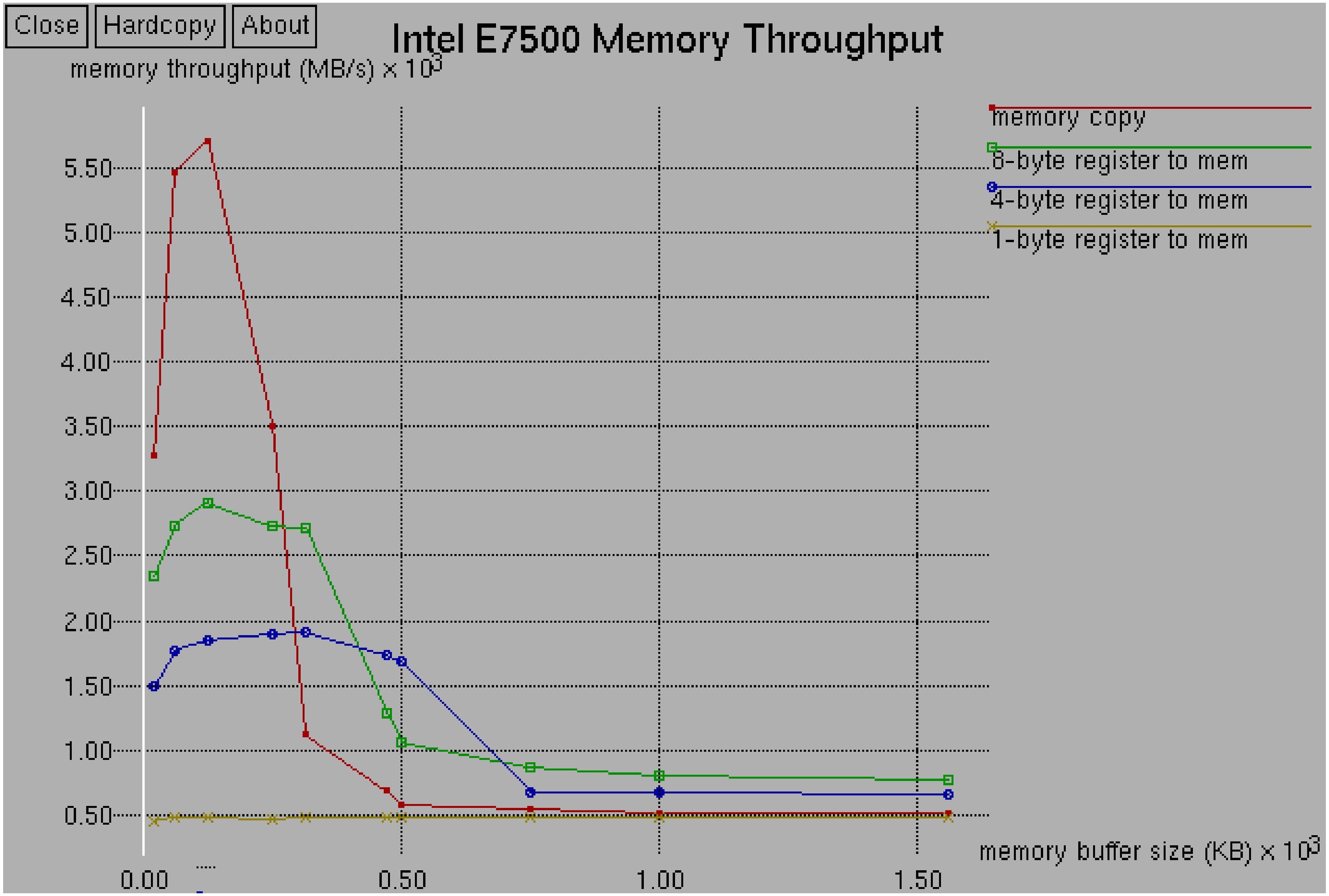
Memory Bandwidth Test



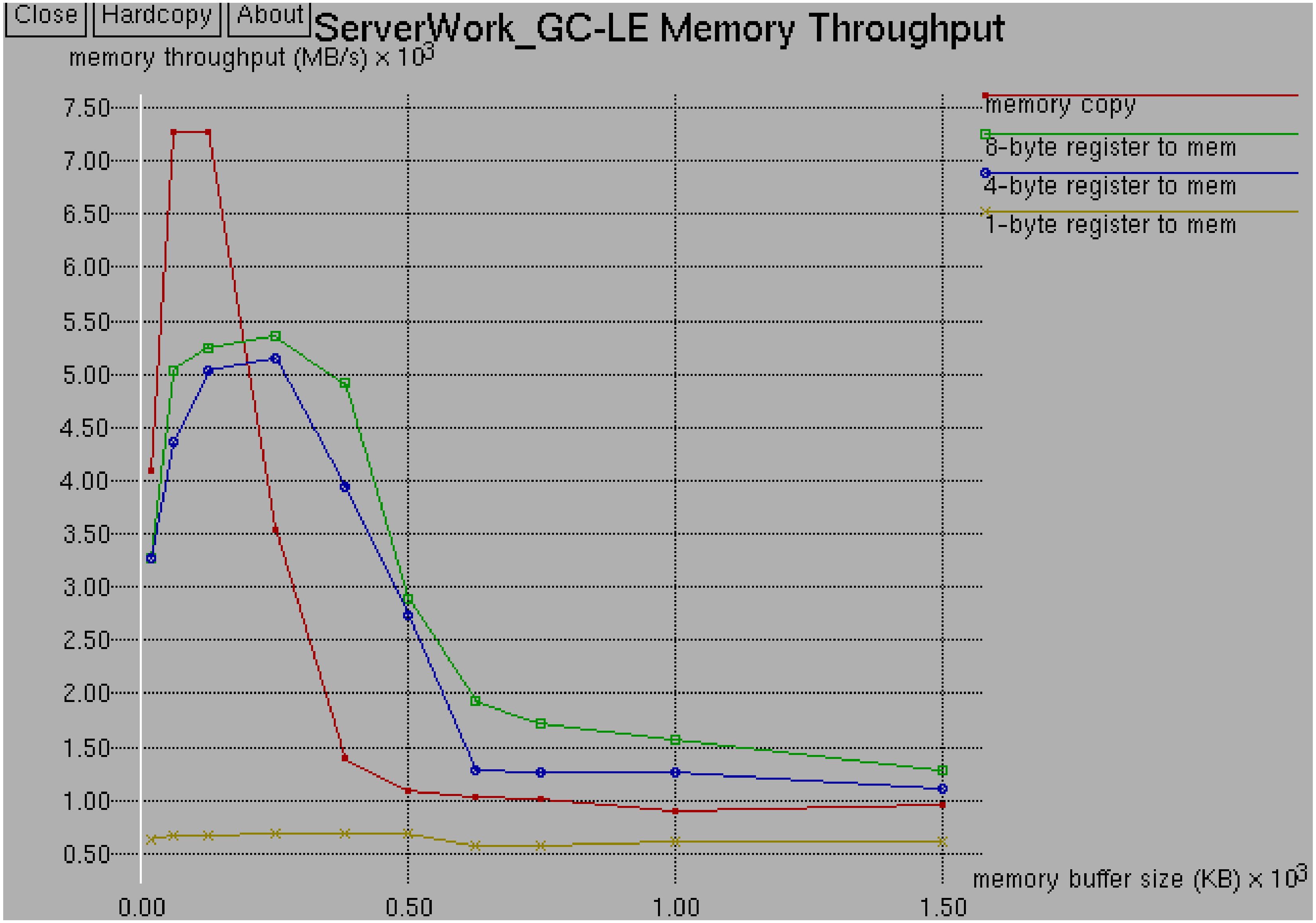
Zoom in Cache Area



More Memory Bandwidth



More Memory Bandwidth



Hardware Bandwidth Abstract

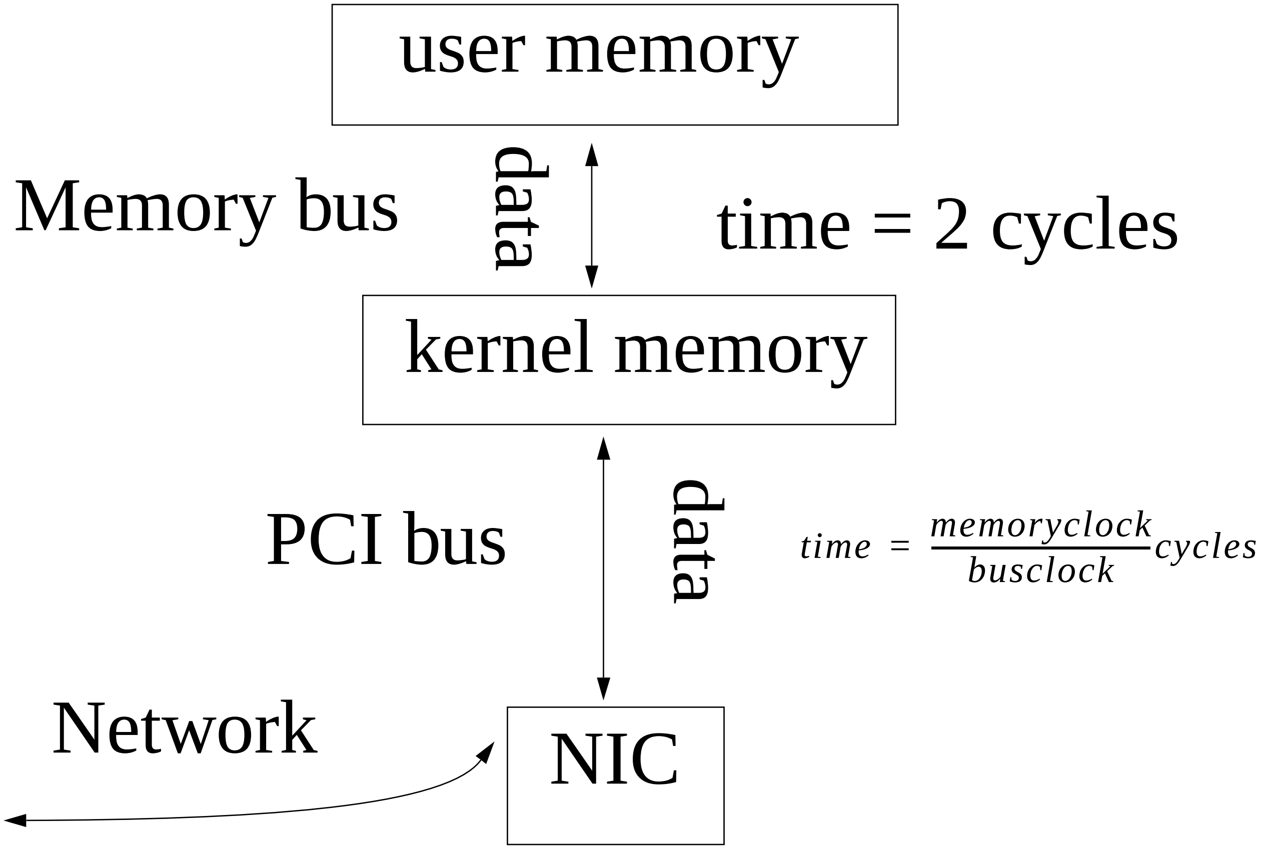


Fig. 1   Hardware data path and Memory cycles for Rx/Tx packets

A Real Case

$$IOthroughput = \frac{MemoryBandwidth}{(PCI + Memory \times 2)cycles} = \frac{MemoryBandwidth}{\frac{MemoryClock}{IOBusClock} + 2} \qquad (1)$$

Both VIA PCI controllers are 32-bit/33MHz.
Year 2000 PCI controller (VIA 868) has 133 MHz memory bus, and it produces 288 MB/s memory bandwidth. The maximum I/O throughput on this system is:

$$IOthroughput = \frac{288}{\frac{133}{33} + 2} \times 8 = 384 \quad \text{Mb/s}$$

2003 VIA PCI controller (VT400) has 400 MHz memory bus, and produces 652 MB/s memory bandwidth. The maximum I/O throughput on this system is:

$$IOthroughput = \frac{652}{\frac{400}{33} + 2} \times 8 = 369 \quad \text{Mb/s}$$

Direct Memory Access (DMA)

Table 1: PCI Burst Size affects DMA performance (32-bit/33 MHz PCI)

Burst Size	Total Bytes Transferred	Total Clocks	Transfer Rate (MB/s)	Latency (ns)
8	32	16	60	480
16	*64*	*24*	*80*	*720*
32	128	40	96	1200
64	256	72	107	2160

Total_Transfer_Clock = 8 + (n - 1) + 1 (Idle cycle)

n is the number of data phases (transfers) per burst
8 is the overhead (maximum) of REQ/IRDY/TRDY

Zero Copy

Only eliminates memory copy between **user** and **kernel** space.
Does not zero out I/O memory copy (DMA)

- It helps if I/O bus speed is close memory bus speed
- It helps to reduce CPU usage, but Must be page aligned

$$percentage = \frac{newThroughput - oldThroughput}{oldThroughput}$$

$$= \frac{2 \times IOBusClock}{MemorySpeed} \qquad (2)$$

$$percentage = \frac{66 \times 2}{133} = 99.2\%$$

$$percentage = \frac{66 \times 2}{400} = 33\%$$

Time Resolution

Table 2: Time of Syscall

timestamp is via gettimeofday API and kernel TSC (microtime)		Linux 2.4.1x		FreeBSD 4.8-RELEASE	
		timestamp ns	read/write ns	timestamp ns	read/write ns
P4 Xeon Intel P4	2.4 GHz	900	1400	4409	1206
	2.0 GHz	980	1100	4590 (3567)	130
MP AMD XP	1730.73 MHz			4195 (4033)	217
	1.4 GHz	282	506		
Intel P4	1.4 GHz	1313	1522		
Intel P3	746.17 MHz	943	2100	4700	289
	531.83 MHz	970	2050	1800 4.3-R	380 4.3-R

More System Call

Table 3: Syscall time for more O.S.

	gettimeofday	read/write
Solaris 2.8 333MHz Sparc	348 ns	8400 ns
Solaris 2.7 400MHz Sparc	278-295 ns	5300 ns
AIX RS 6000	> 3000 ns	8500 ns
IRIX 2.6 175 MHz IP28	7946 ns	28162 ns
BSD/OS 526 MHz PII	10877 ns	11357 ns
Mac OS X 1GHz G4	1937 ns	2043 ns

If copying 20 KB data from user space to NIC takes 100 µs and each write syscall is 1 µs:
Sending one 20-KB datagram will use 101 µs
and
Send 20 1KB datagrams needs 120 µs

I/O Interrupt

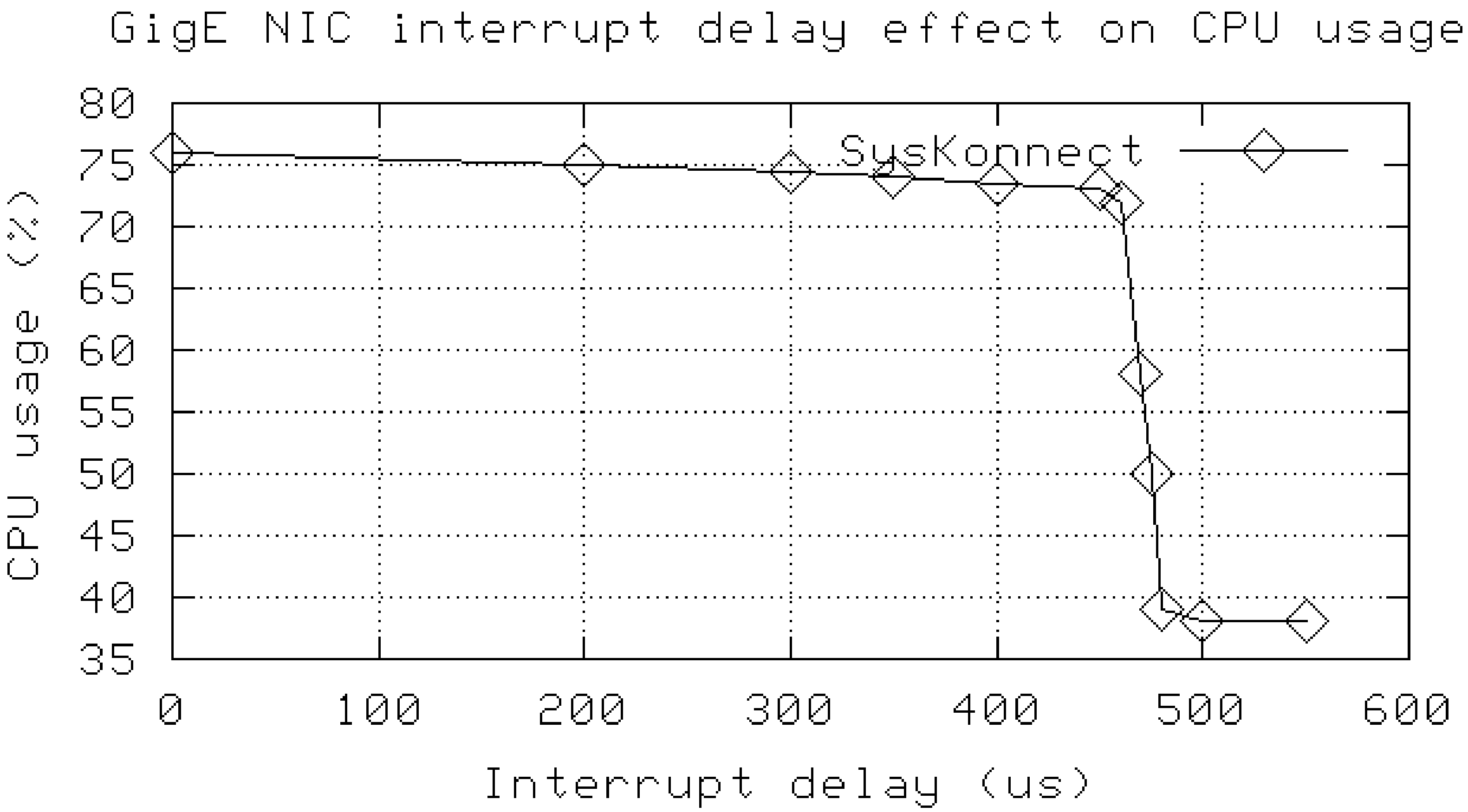
Table 4: CPU utilization affected by I/O interrupt

interrupt delay time (coalescing)	% CPU IDLE	% CPU Interrupt	Throughput Mb/s
64 µs interrupt delay for Intel 82540 copper GigE (PCI/66) + Intel P4 Xeon 3 GHz CPU	0	92	277
300 µs interrupt delay for above configuration	1	72	515

Cheap copper GigE NIC chews all high-end CPU

- Issues of doing interrupt delay:
  - non linear tuning feature
  - unknown packet arrival time

I/O Interrupt Continued



CPU Clock Counter (CCC) and Device Onboard Timestamp (TSC)

CCC and TSC may provide 100-time higher time resolution then gettimeofday

Issues:

- How to deliver it from kernel to user space?
- How to get CPU or controller clock rate because CCC and TSC are counters, not time.
- They may still need to get system timer for reference.
- They need to modify device driver and other kernel source files.
- Availability

Onboard Timestamp

Get_SystemTime - (Get_CurrentNIC_ClockCounter - packet_TimeCounter)

- Work around I/O interrupt coalescing (moderation) ✓
- Improve clock resolution (Just to compare with system timer) ✗
  - Still needs to access CTC (TSC) to get system timer
  - Needs 1~3 reads to get onboard clock counter — this will use I/O bus:

(1) AMD 1.67GHz MP + Tyan S2466N
Takes minimum 61 tick counters of 31.25 MHz clock
equivalent to 1.952 μs (1951 ns)
Average 69 ticks = 2.2 μs

(2) Intel 2.0 GHz Xeon + Supermicro P4DPE
Minimum 131 ticks = 4.192 μs
Average 137 ticks = 4.384 μs

```
static __inline nic_ts_t
sk_read_CTC(register struct sk_if_softc *sc_if)
{
register nic_ts_t lo, hi;

        hi = SK_XM_READ_2(sc_if, XM_TSTAMP_READ + 2) << 16;
        lo = SK_XM_READ_2(sc_if, XM_TSTAMP_READ) & 0xFFFF;
        if (lo < 1562)  /* SK_XM_READ_2 should never exceed 50 us.      */
                hi = SK_XM_READ_2(sc_if, XM_TSTAMP_READ + 2) << 16;
return (hi | lo);
}
```

Issues to use on board Time Stamp:

(1) Most vendors do not provide it.

(2) It requires to modify the device driver as well as either BPF or network stack to pass timestamp to user level.

Storage System

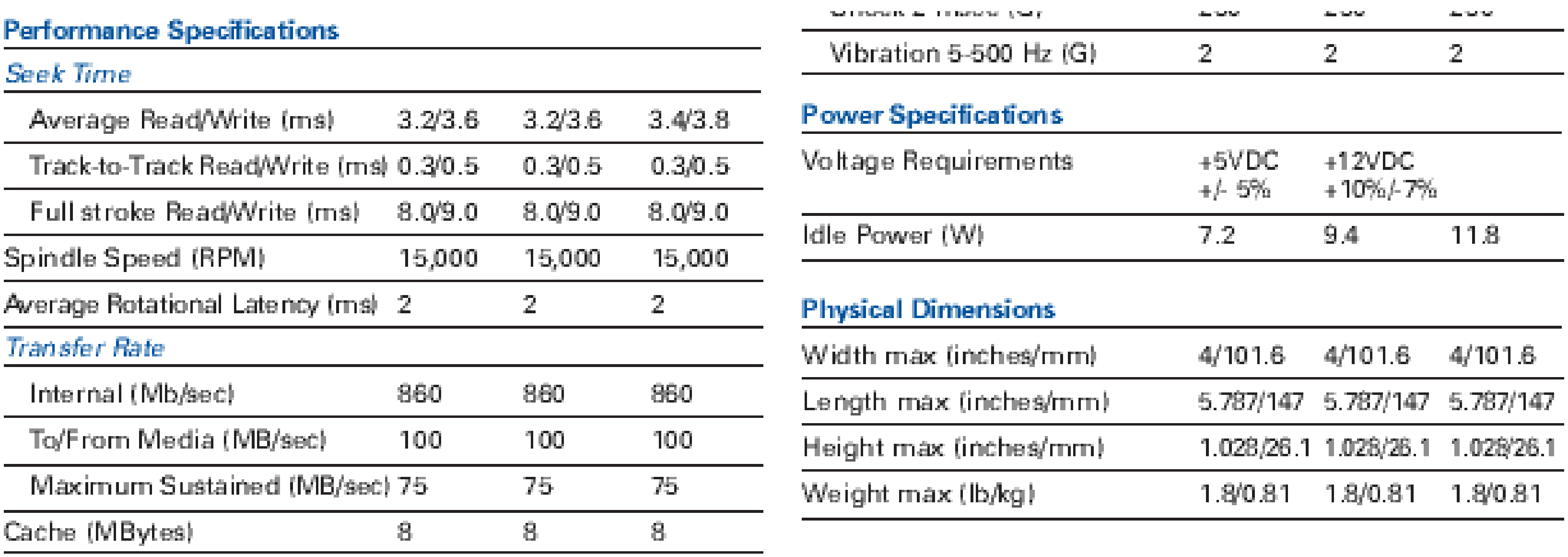
No more raw device under new UN*X

- Disk controllers and Disk drives (same performance)
  - SCSI — MTBF   1,200,000 hours;   Service Life   5 years
    - 15 drives per bus
  - IDE — MTBF   680,000 hours (77.6 years), but less than 1/2 price
    - 2 drives per bus

- Raid (redundant array of independent disks; originally redundant array of inexpensive disks)
  - RAID-0   stripping to increase performance
  - RAID-1 or RAID-5   increases the mean time between failure (MTBF), storing data redundantly also increases fault-tolerance

- Tape — cheap, slow, but more reliable

Maxtor drive’s data sheet



Performance Specifications

Seek Time			
Average Read/Write (ms)	3.2/3.6	3.2/3.6	3.4/3.8
Track-to-Track Read/Write (ms)	0.3/0.5	0.3/0.5	0.3/0.5
Full stroke Read/Write (ms)	8.0/9.0	8.0/9.0	8.0/9.0
Spindle Speed (RPM)	15,000	15,000	15,000
Average Rotational Latency (ms)	2	2	2
Transfer Rate			
Internal (Mb/sec)	860	860	860
To/From Media (MB/sec)	100	100	100
Maximum Sustained (MB/sec)	75	75	75
Cache (MBytes)	8	8	8

Vibration 5-500 Hz (G)	2	2	2

Power Specifications

Voltage Requirements	+5VDC +/- 5%	+12VDC +10%/-7%	
Idle Power (W)	7.2	9.4	11.8

Physical Dimensions

Width max (inches/mm)	4/101.6	4/101.6	4/101.6
Length max (inches/mm)	5.787/147	5.787/147	5.787/147
Height max (inches/mm)	1.028/26.1	1.028/26.1	1.028/26.1
Weight max (lb/kg)	1.8/0.81	1.8/0.81	1.8/0.81

WDC drive's data sheet

Data Transfer Rate (maximum)

- Buffer to Host

  100 MB/s (Mode 5 Ultra ATA)
  66.6 MB/s (Mode 4 Ultra ATA)
  33.3 MB/s (Mode 2 Ultra ATA)
  16.6 MB/s (Mode 4 PIO)
  16.6 MB/s (Mode 2 multi-word DMA)

- Buffer to Disk

  748 Mbits/s maximum —> 93.5 MB/s
  478 Mbits/s minimum —> 59.75 MB/s

So, the average bulk transfer can be calculated:
(93.5 + 59.75) ÷ 2 = 76.625 MB/s

Combined I/O Bandwidth

Read data from disk and send it to remote host via network, the maximum I/O traffic must be designed to balance with all I/O requests.

- 1 Gb/s network:
  - IDE bus is about 100 MB/s (0.8 Gb/s)
    - AMD 760 and nVidia IDE controller can go 200 MB/s (1.6 Gbs)
  - SCSI bus is about 320 MB/s (2.56 Gb/s)
    - Single disk I/O is 60 MB/s (480 Mb/s) — not bottleneck
    - Single disk I/O is 36 MB/s (288 Mb/s) — the bottleneck

Stripping the disk to increase the disk I/O rate which single disk I/O rate is the bottleneck.

Future I/O bandwidth

- PCI-X 2.0
  - 66 and 133 MHz (3.3v) — available now 528~1064 MB/s (8.5 Gb/s)
  - 266 and 532 MHz (1.5v) — 2128~4256 MB/s (34 Gb/s)
- PCI-X 3.0 (likely will be bypassed because of distance)
  - 1066 and 2133 MHz (1.5v) — 8.5~17 GB/s (136 Gb/s)
- InfiniBand (IB - NGIO) and Host Channel Adapters (HCA)
- PCI-Express (3GIO) — Semi Asynchronized
  - 1, 2, 4 ... 32 lanes (2.5Gb/s each, 2.0 effective rate — 64Gb/s total)

Note: realize standard to practice needs time.

Context Switch

- The context switch period is 10 ms
  - usleep(1) == usleep(10,000)
  - Use usleep() or select() for delay time in multiple context switch (10ms)

- To avoid context switch effect
  - design each measurement process in less than context switch period
  - voluntarily switch the process out before start each measurement

CPUs and SMP kernel

Symmetric Multiple Processor (SMP) technology provides more CPU power on a single machine.

- Advantage:
  - 64-bit / 66~133 MHz PCI/PCI-X bus are most likely only supported on SMP motherboards
- Disadvantages:
  - Plug in a second CPU without using it can reduce 10-15% bus bandwidth
  - Multiple CPUs plus SMP kernel will reduce bus bandwidth 10-15% more
- Dual CPUs do not mean double computation power

Cost-Effective factor:

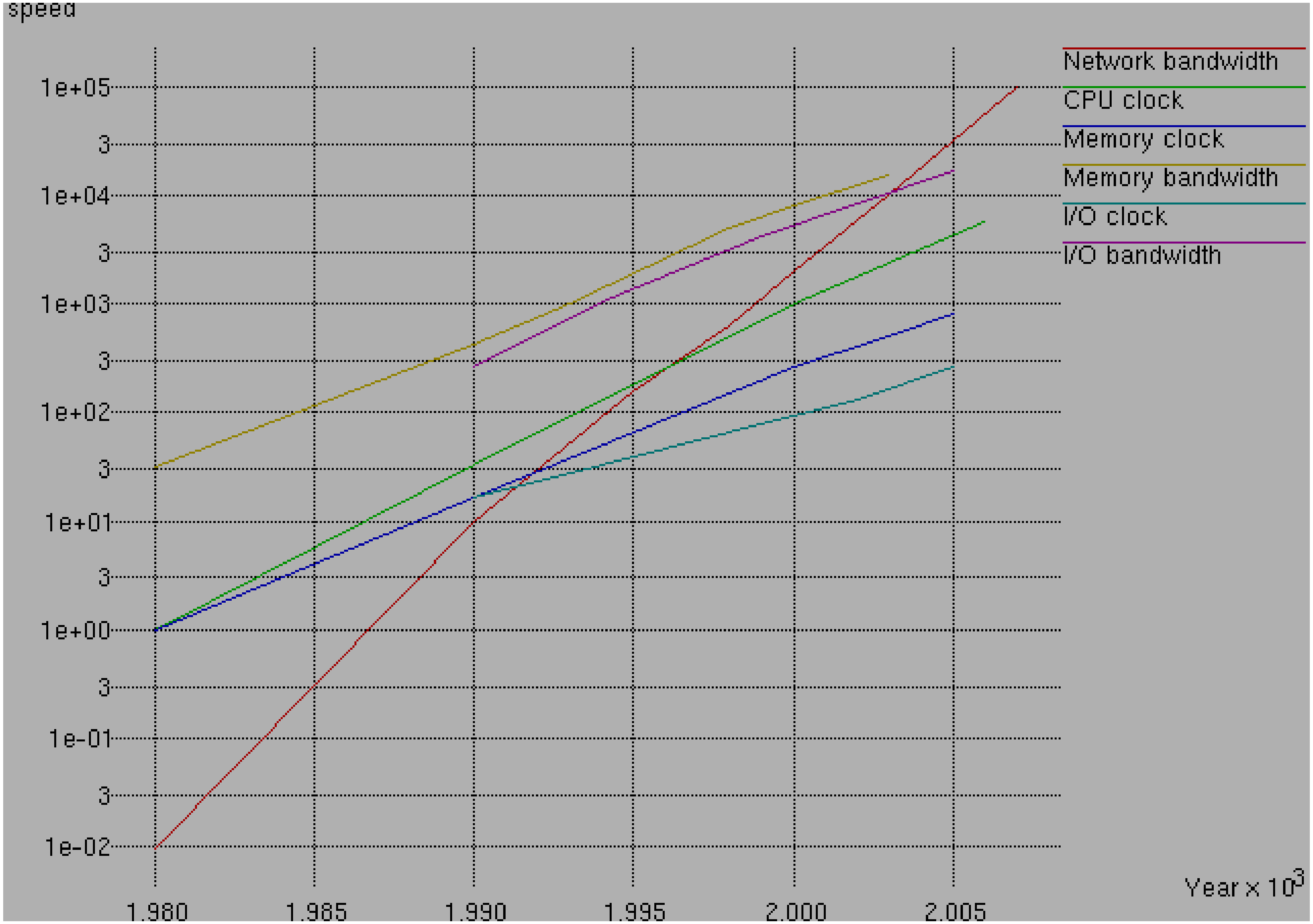
CPU: The one generation older one is 1/2 price of the latest one.

Disk Drivers by end of Oct. 2003

SCSI:	below 73 GB is 0.85~1GB/per $ 73 GB is about 0.5 GB/per $ 143 GB is about 0.3539 GB/per $ 181 GB is about 0.2265 GB/per $
EIDE:	0.88 GB/per $
SATA:	0.96~1.2 GB/per $
Firewire:	1 GB/per $

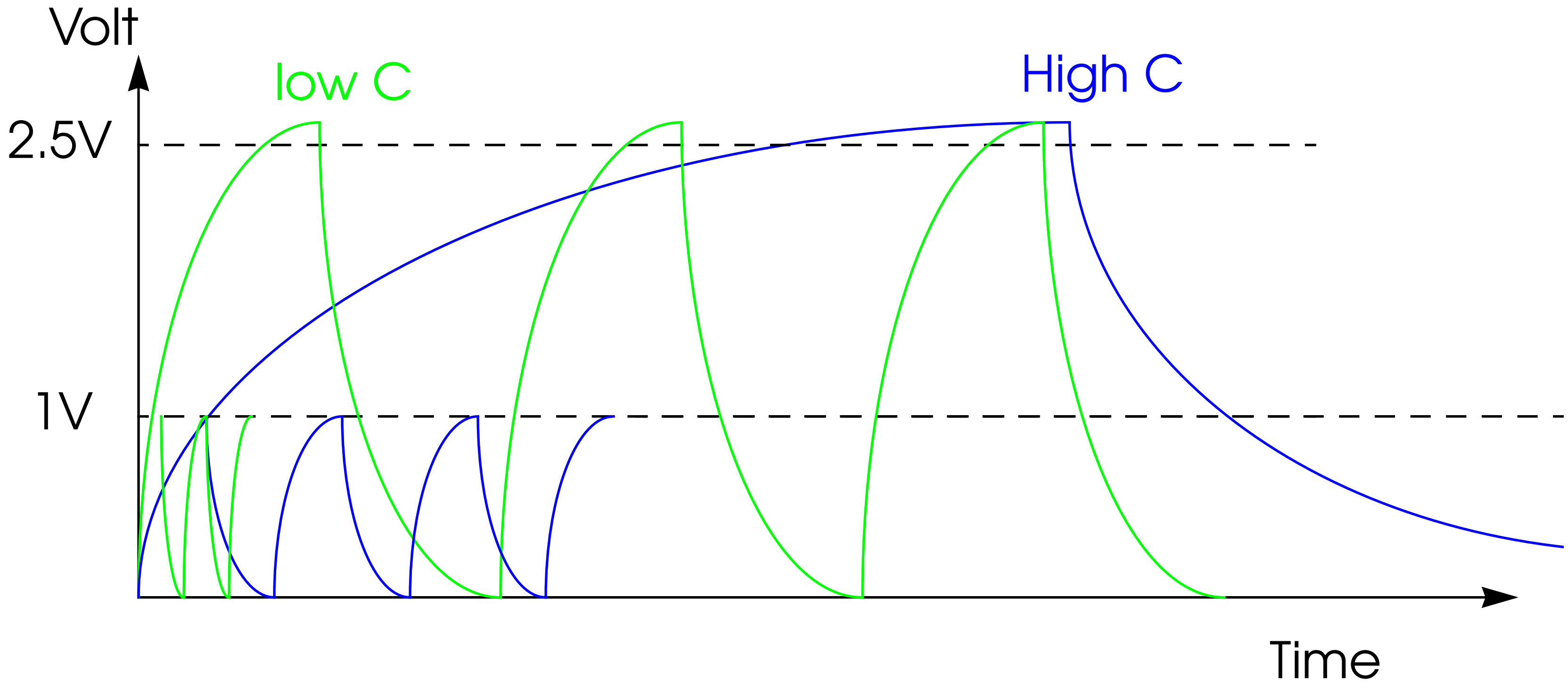
If you do not need to use hardware right way, wait till you use it. Price changes every quarter.

Speed Evolution of Each SubSystems

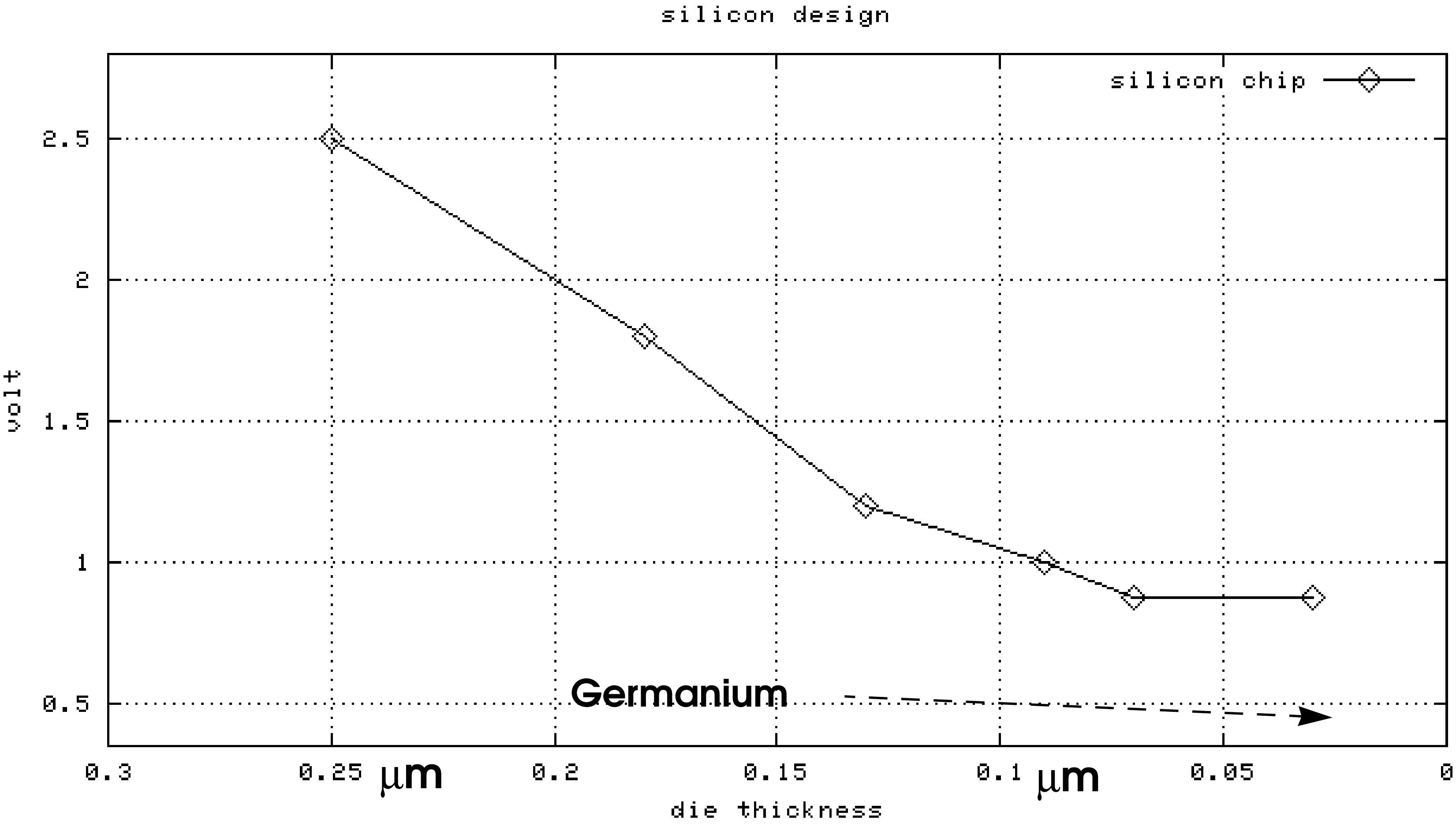


Design and Determine CPU Speed

The thinner the die is, the lower the voltage and capacitor will be



Chip Design



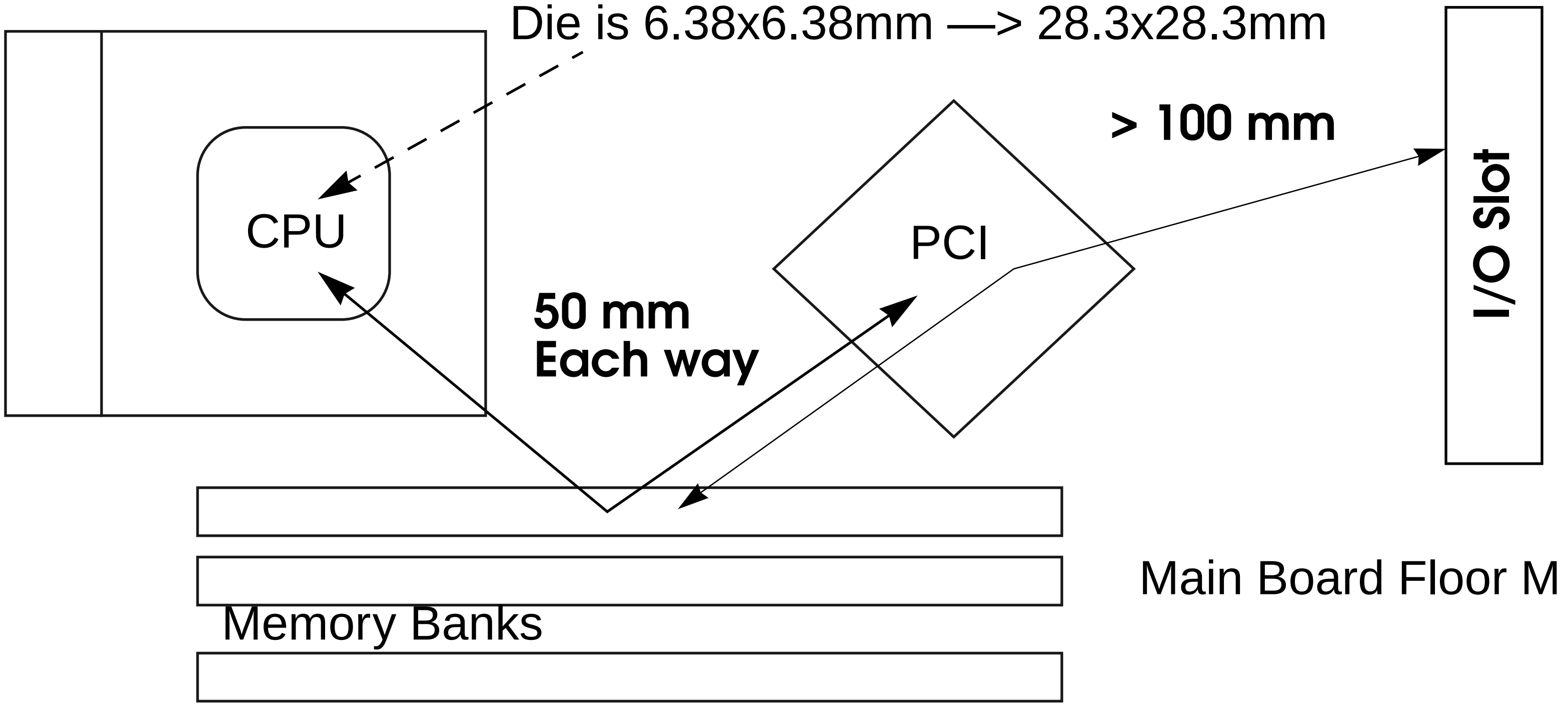
Further lower the voltage, static power will be higher than active power

How Far the Silicon can GO

(1) Light speed is 300,000 Km/sec = 300mm/ns

10GHz clock —> 0.1 ns per clock cycle, light can travel 30mm per clock cycle

With 100GHz clock, light can travel only 3 mm per clock cycle



Die is 6.38x6.38mm —> 28.3x28.3mm

CPU

PCI

> 100 mm

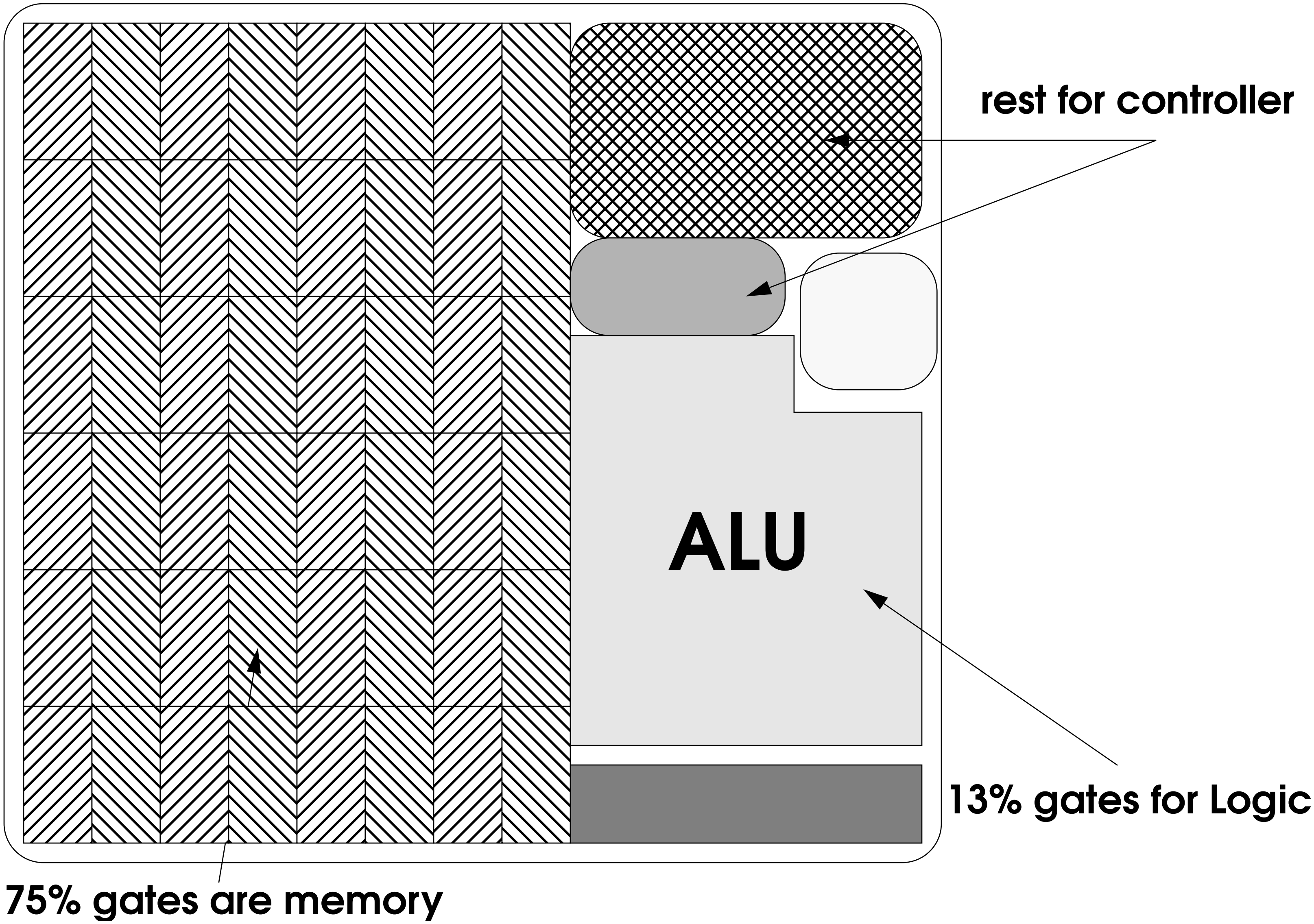
I/O Slot

50 mm
Each way

Memory Banks

Main Board Floor Map

CPU layout



rest for controller

13% gates for Logic

75% gates are memory

Network Interface Adapter

R. Hughes-Jones Manchester

PCI: SysKonnect SK-9843

Motherboard: SuperMicro 370DLE Chipset: ServerWorks III LE Chipset

CPU: PIII 800 MHz **PCI:64 bit 66 MHz**

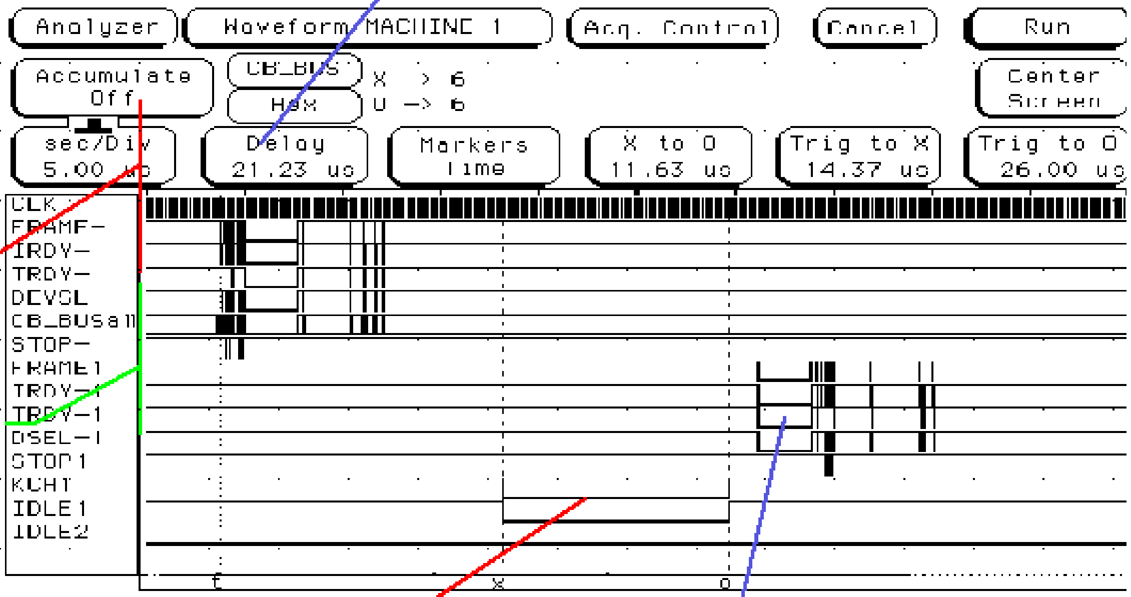
RedHat 7.1 Kernel 2.4.14

SK300

1400 bytes sent

Wait 100 us

~8 us for send or receive



sec/Div 5.00 us

Delay 21.23 us

X to O 11.63 us

Trig to X 14.37 us

Trig to O 26.00 us

Send data mem ? NIC

Send PCI

Receive PCI

Gigabit Ethernet frame 11.6 ?s

Receive data NIC ? mem

Understanding Algorithms — Architecture Design Issues

- Single packet — physical bandwidth
  - pathchar
- Packet pair, Packet Quad — physical bandwidth
  - nettimer
- Spaced Packets — available bandwidth
  - Constant spacing
    - pathload, igi, etc.
  - Variable spacing
    - pathChirp
- Packet Train — bandwidths, achievable throughput, MBS, etc.
  - NCS, netest

Variable Packet Size

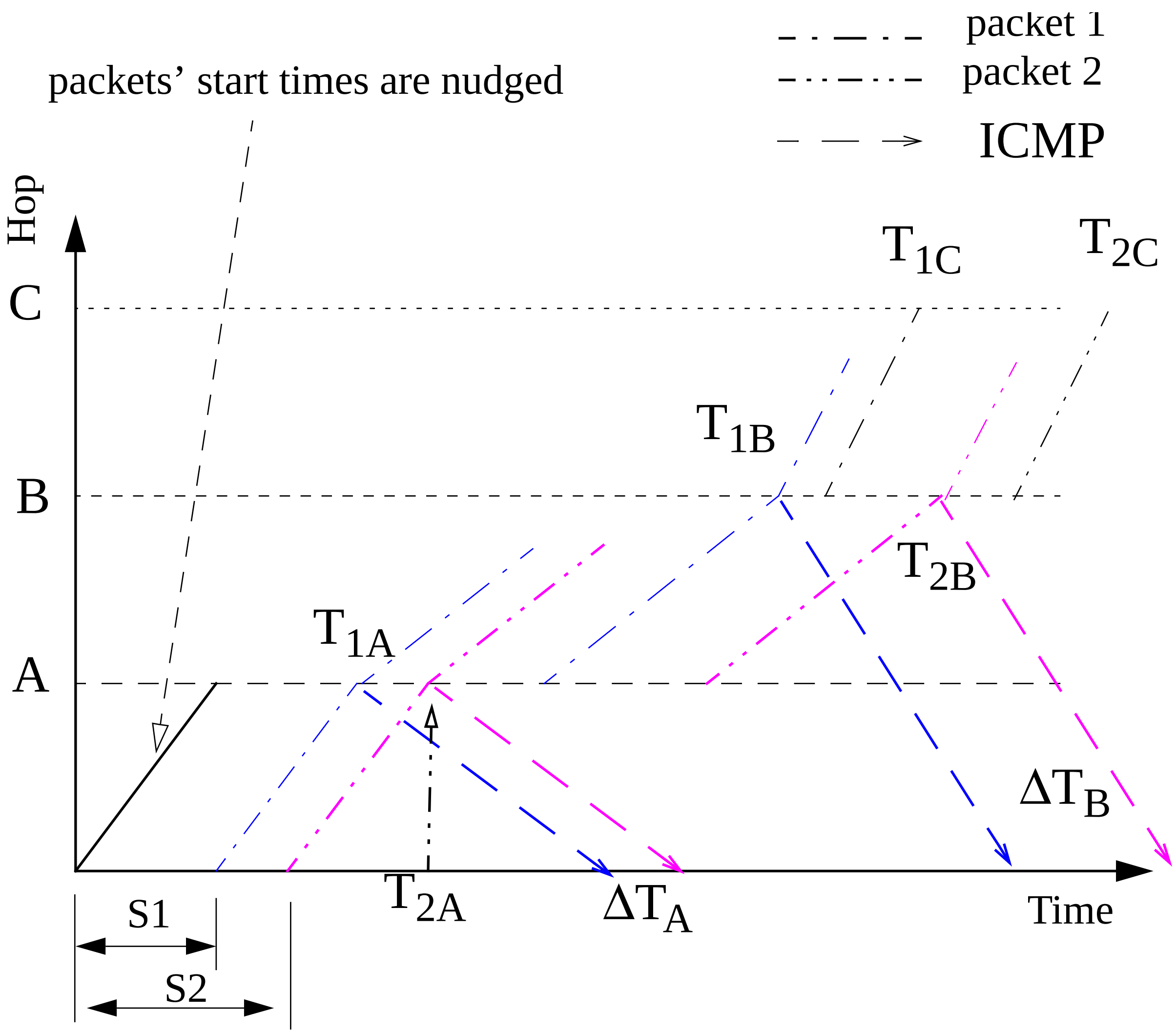


Figure 2. VPS transfer timing of two packets on a network route

VPS Continued

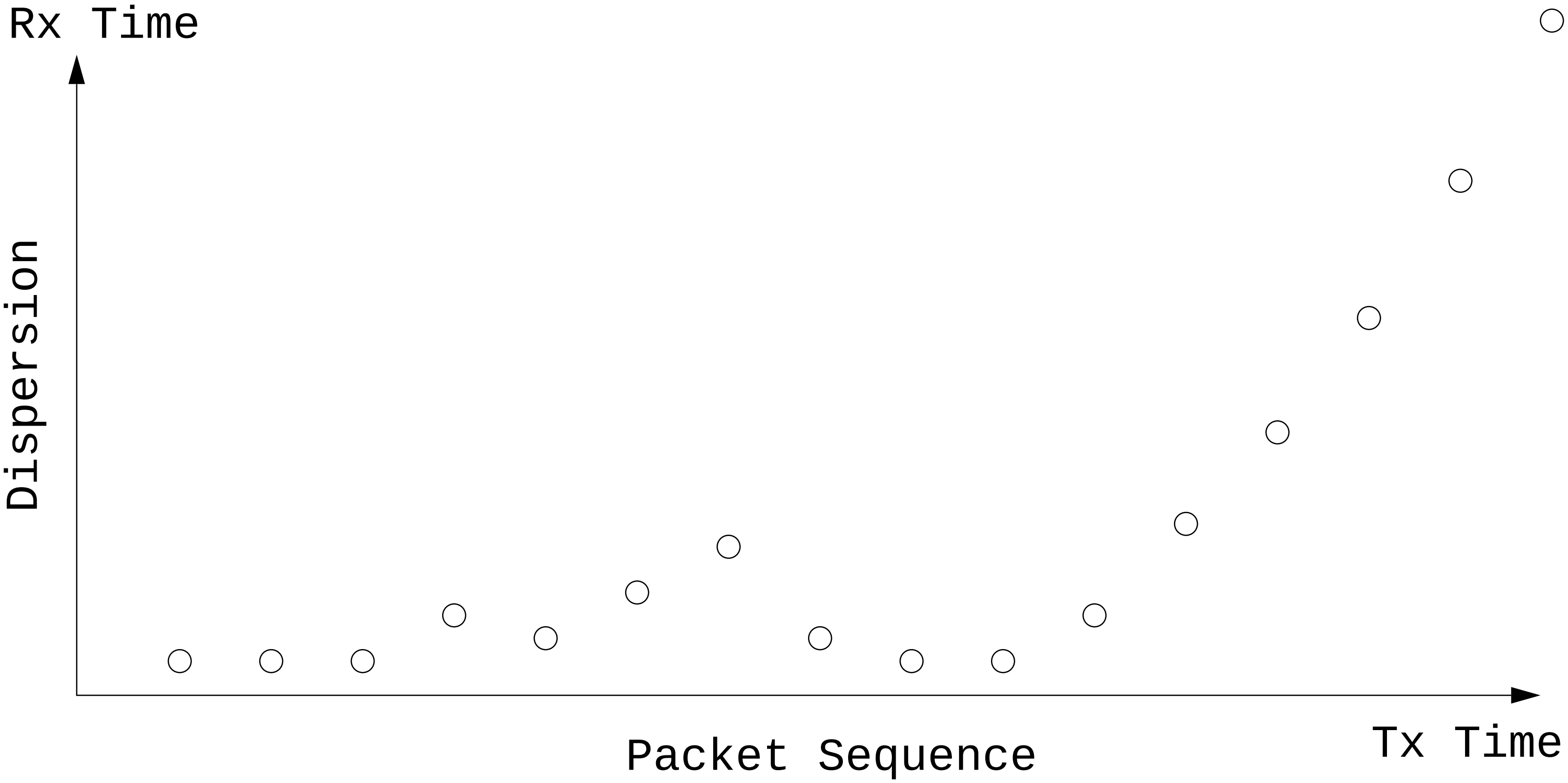
Maximum size difference = 1472 byte (Fat pipe <= 1 Gb/s)

Typical RTT fluctuation 50~300 μs

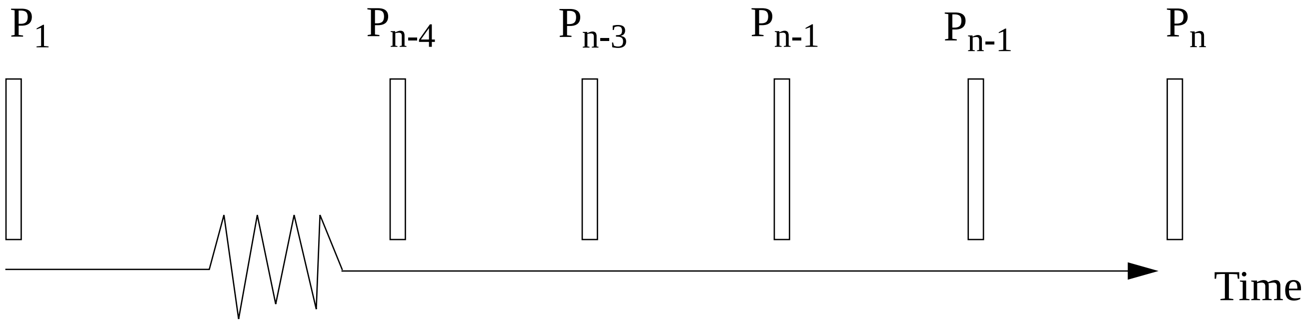
Using minimum RTT fluctuation (50 μs) with maximum 50% error, the minimum measurable RTT difference is 100 μs:

$$MaximumMeasureRate = \frac{1472 \times 8}{100\mu s} = 117.76Mbps$$

Packet Pair(s) and Dispersion



Constant Spacing

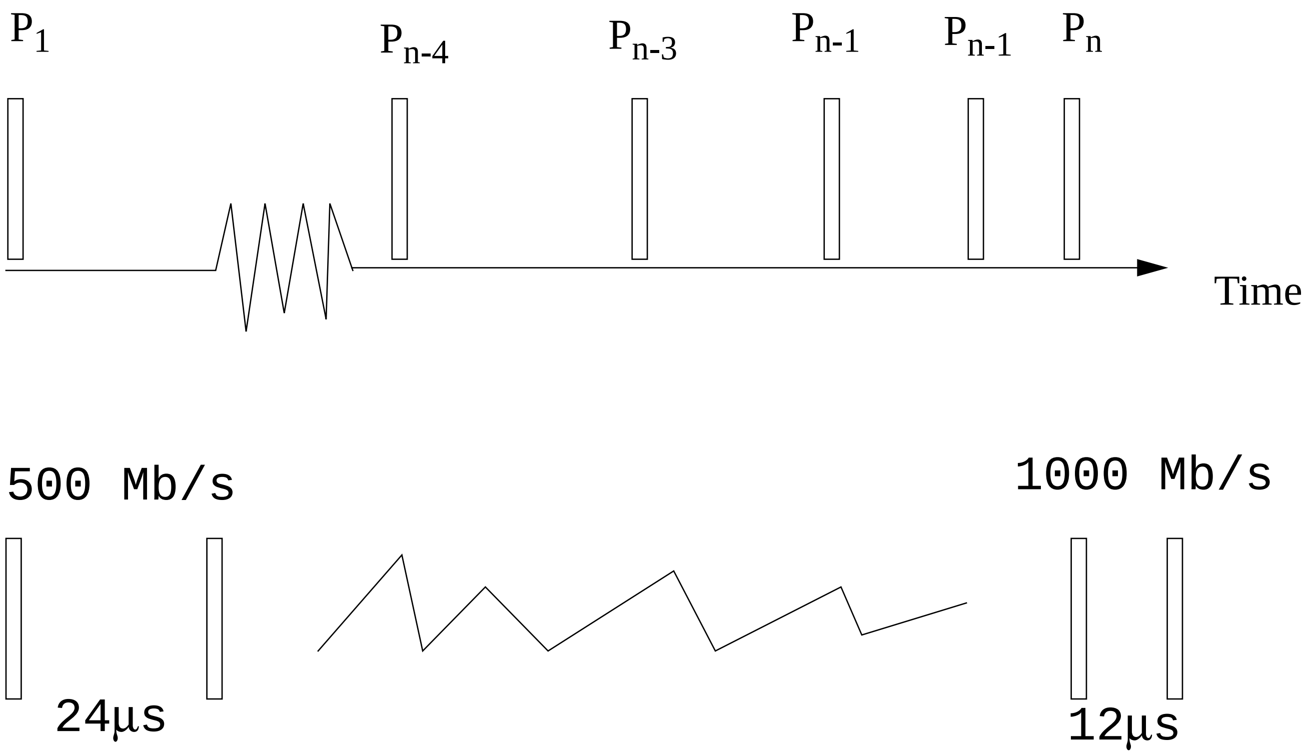


1500-byte packet over 1 Gb/s = 12 µs

1500-byte packet over 10 Gb/s = 1.2 µs

- I/O interrupt coalescing bunches them together on high-speed NICs
- Spacing measurement — 1.2 µs~12 µs

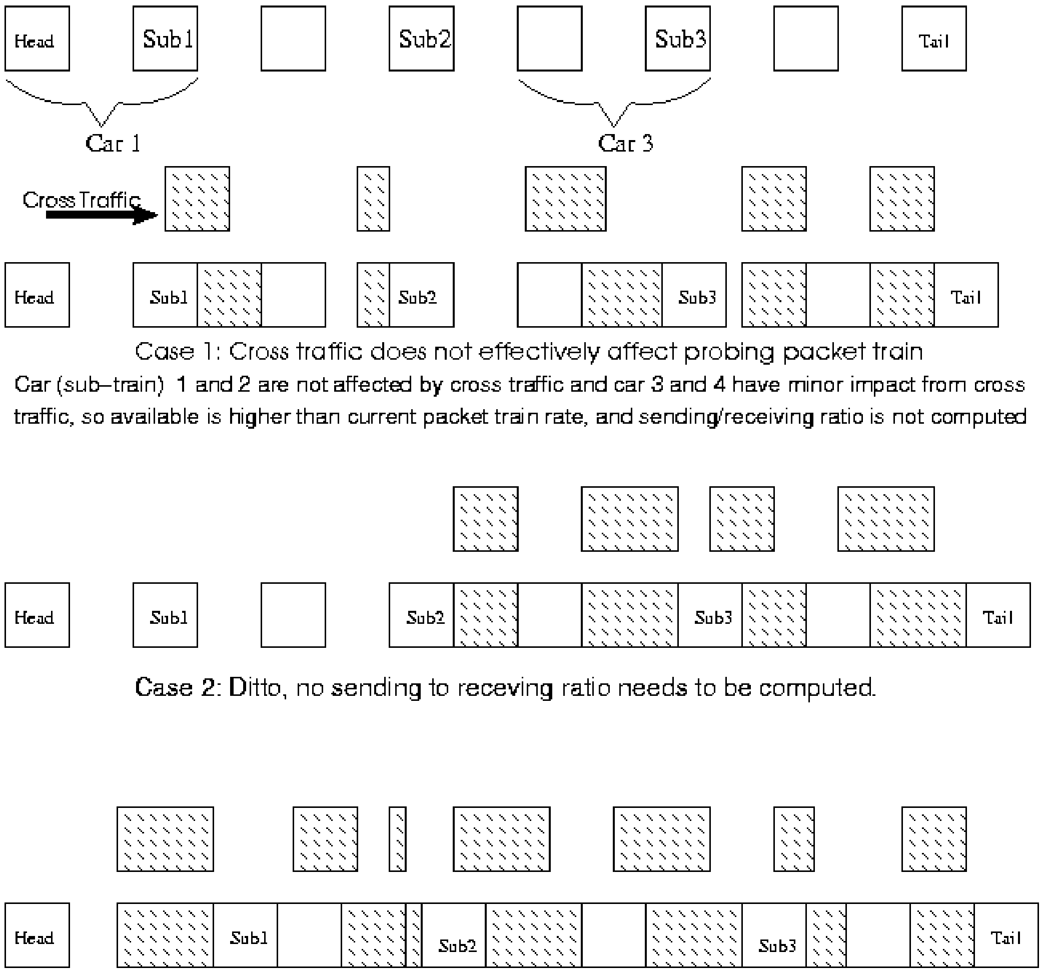
Variable Spacing



- A limited packets can be sent in this period
- Any cross traffic can affect the spacing (under estimation of the bandwidth)
- I/O interrupt coalescing bunches they together (over estimation)

Packet Train

Original packet train sent out from source NIC for probing



Case 1: Cross traffic does not effectively affect probing packet train

Car (sub–train) 1 and 2 are not affected by cross traffic and car 3 and 4 have minor impact from cross traffic, so available is higher than current packet train rate, and sending/receiving ratio is not computed

Case 2: Ditto, no sending to receving ratio needs to be computed.

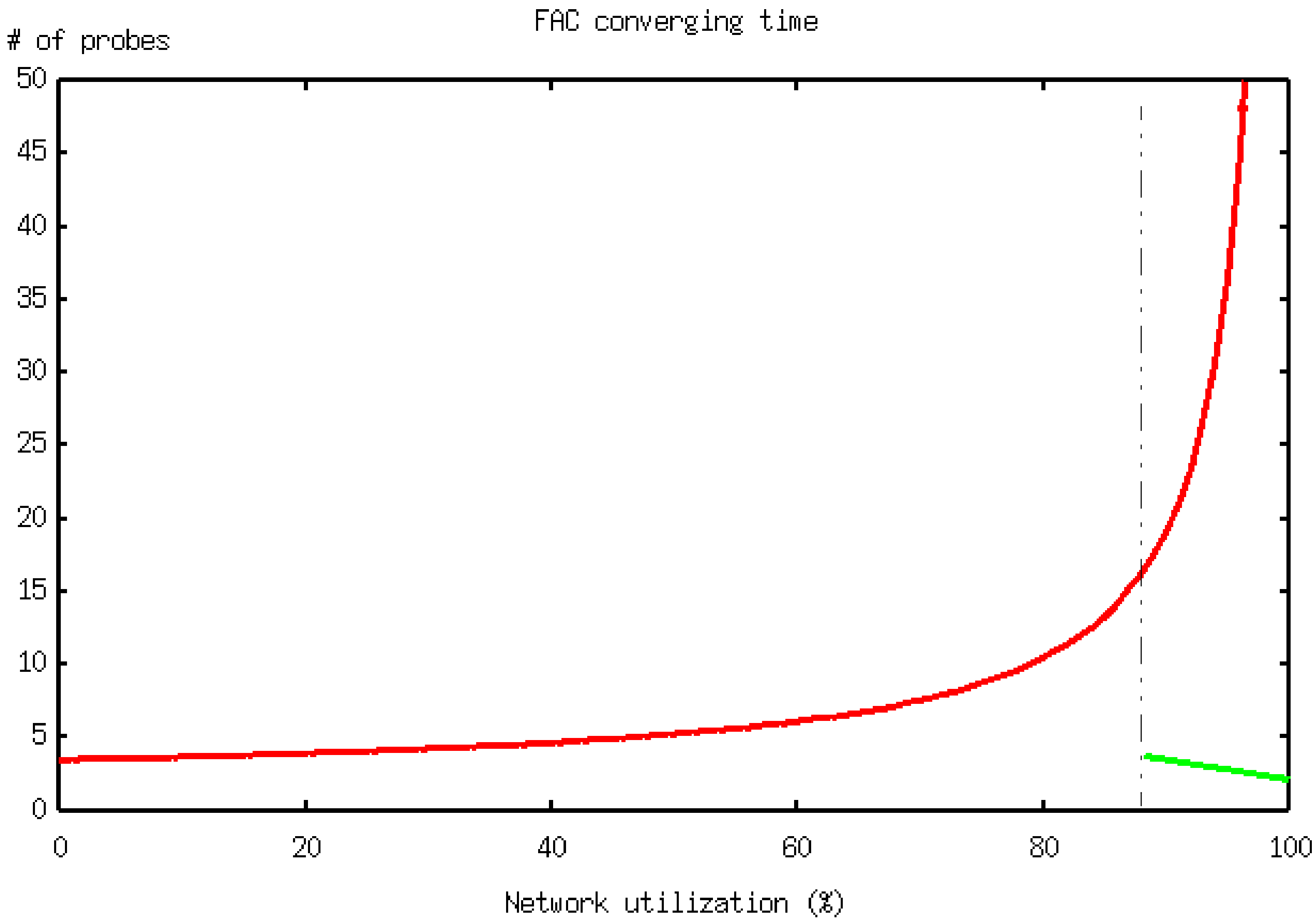
Case 3: able to detect cross traffic

All sub trains are affected by cross traffic as well as the main train. and sending/receiving ratio is computed.

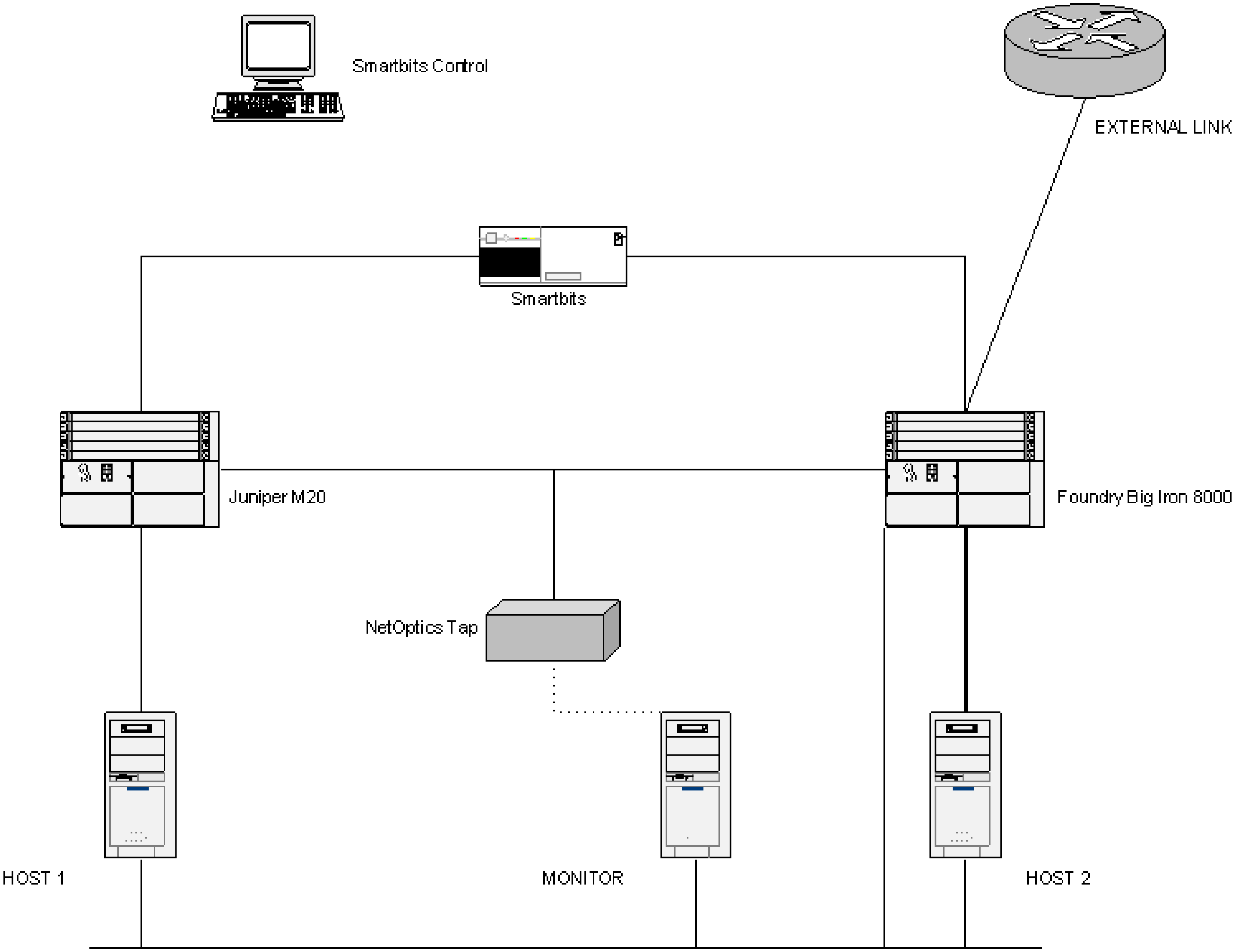
- Complicated than packet spacing

Car —
a measurement UNIT consists of a set of multiple MTU packets

FAC^2 Converging Time in Practice



CAIDA Emulation Network Topology



Emulation testbed results

Utilization % (loss %)	**run time (sec.)**	**netest results**	**Accuracy**
GigE network MTU = 9K 50~100 tests per run (300 sec.)	(*require longer measurement duration*)	**available bandwidth (Mb/s)**	**(%)**
0 (0)	**2.4 - 6.5** including MBS measurement	maximum throughput: 851	
10 (0)			
20 (0)		791.0 - 791.2	98.875
30 (0)		690.0- 691.0	98.643
40 (0)		598.5 - 599.0	99.750
50 (0)		502.5 - 502.9	99.420
60 (0)		403.8 - 403.9	99.025
70 (0)		306.4 - 306.6	97.833
80 (0.01)	*7.89* *(11.9)*	210.0 - 211.0 205	94.500 97.500
90 (0.01)	13-*15* *(26)*	113.0 - 115.0 102	86.000 98.000

Conclusion

- Systematic engineering design is critical for:
  - exploring measurement algorithms
  - implementing accurate tools
  - building high performance applications (software)

More information

http://dsd.lbl.gov/NCS